This application is submitted in the names of John C. Gaddy and John D. Marshall and has been assigned to Timbral Research Inc.

SPECIFICATION

SYNCHRONIZED STREAMED PLAYBACK AND RECORDING FOR PERSONAL COMPUTERS

FIELD OF THE INVENTION

The present invention relates to the recording of audio files. More particularly the invention relates to the synchronization of an audio capture device and an audio playback device on a personal computer.

PRIOR ART

In order to create an original "master recording" of a musical ensemble performance a recording engineer must be employed. A recording engineer is responsible for running the mixer, a device that allows the recording engineer to adjust the volume levels and tone of each musical instrument to create a balance of

09755623 . 022701

the instruments that is aesthetically pleasing. Additionally, the mixing board allows a recording engineer to combine a multi-track recording into a two track stereo master recording. For example, a simple recording of a garage band may have four tracks, one track for the lead vocal, a second track for guitar, a third track for the bass, and a fourth track for drums. By recording each of the instruments on a separate track, the volume levels and tone of each instrument can be adjusted separately to fine-tune the overall sound of the recording. Then, after the levels are adjusted, the four-track recording is mixed down to a two track stereo recording.

In order to make a multi-track recording, the sound engineer must make sure that the timing of each track is exactly synchronized with each other. If one track is not exactly in synch with the others, then the finished recording will have an undesirable sound due to the misalignment of the tracks. For example, the bass may be out of synch with and therefore interfere with the vocalist.

Additionally, in order to make a multi-track recording, an artist must have the necessary equipment or hire a recording engineer who has the necessary

equipment to make the recording. Furthermore, the recording engineer must be paid for services. Many artists cannot afford to hire a recording engineer every time they wish to make a recording. For example, artists may wish to make a recording so that they can review their performance. Additionally, an artist may wish to make a recording of a practice session so that the artist can then replay his or her performance for a music teacher to critique.

Personal computers running Windows® software have Windows Media Player included in the operating system. When this simple audio recording program is coupled with a sound card, it provides one-track audio recording capability using an inexpensive microphone.

If the user is unsatisfied with the one track recording provided by pre-installed programs such as Windows Media Player, then the user may choose a multi-track program that is compatible with Windows® DirectSoundCapture capabilities. Although these programs allow users to capture multi-track audio on a computer, using them requires much skill in the art. Extensive and complex instruction manuals are included, and technical support is often required after

users study the manuals. Then experience with the software is needed before users become proficient enough to make a recording.

With the advent of the Internet, an artist could log onto the Internet and find downloadable accompaniment over which the artist could record a part using multi-track software. Using this method, a musician would not need to assemble a complete band or be a multi-instrumentalist in order to make a multi-track recording. However, the files containing this accompaniment would be extremely large. For example, if a singer were to download a piano, bass and drum track for a three-minute song, the total size of the stereo file would be 31.8 MB, which would take at least 1 hour and 15 minutes to download over a 56 kbps modem. The download time would be shorter if the singer had a faster, broadband connection to the Internet, such as ISDN or a cable modem. However, many people, especially people in small towns and rural areas, do not have access to broadband services.

To address the problem of the time it takes to download audio files, software programs were developed that allow users to "stream" audio files from a

09750655.122701

server to which they can connect over the Internet or an intranet. The streamed audio is not downloaded onto a client computer. The audio stream player simply receives a signal and plays the files, similar to how a radio plays a signal without capturing it on the user's radio for later playback. Software components such as RealPlayer® produced by RealNetworks, Inc. allow users to play a streamed audio file on their computer at different levels of quality, depending on the speed of their Internet connection. When a digital audio file is streamed over a network, a server sends the file piecemeal over a network to a client computer. The client buffers the incoming data and monitors the delivery rate of the data from the server. If the client software determines that the network bandwidth is insufficient to render the audio file in an uninterrupted fashion, it signals the server to switch over to a lower bandwidth version of the audio data and the streaming and rendering continue uninterrupted. Conversely, if network conditions improve, the client software signals the server to switch to a higher bandwidth stream and a higher quality audio signal is rendered.

As mentioned above, artists may wish to stream an audio file of a musical performance in which they will perform a part. For example, an artist may wish to

practice his or her singing, so the artist would choose to stream an audio file from a server that contains accompaniment. The artist would log onto a network where the artist could gain access to a proprietary server from which the files would be streamed. Then if the artist wished to record his or her song with the streamed accompaniment, the artist would need to connect a microphone to the sound card, which would convert the analog audio captured by the microphone to digital data. The digital data could then be copied (recorded) into a file that would contain the digital audio data for playback and for possible mixing with the streamed accompaniment. However, this method would not provide a means of capturing the streamed accompaniment as an audio file that could be mixed with the singing, so that the artist could play back and review a mix of his or her performance that is synchronized with the accompaniment.

To record both the streamed accompaniment and the song, the artist would need to play the streamed accompaniment and begin to record his or her song with it. To synchronize the accompaniment with the performance, the singer would have to carefully watch the control panel of the audio stream player so that the artist could start the recording at exactly the same moment the streamed

accompaniment began in order to synchronize the performance with the accompaniment; otherwise it would be very difficult, if not impossible, to later mix them into one file so that they could be played back as one integrated stereo or monaural audio file. Streaming the accompaniment would solve the problem of the time it takes to download files before an artist can start practicing them or recording with them, but it introduces the problem of capturing and adequately synchronizing the accompaniment with a vocal or instrumental performance.

Therefore, there is a need for a method that automatically captures an artist's performance while rendering a streamed accompaniment for the artist to perform with. There is also a need for a method that aligns the start time of an artist's performance with the start time of a streamed accompaniment, so that a mix can be made for playback.

Accordingly, the current invention is described which addresses the need for streaming and rendering an audio program and simultaneously capturing an audio performance in such a way that synchronization of the two audio data sets is guaranteed. The current invention may be used in conjunction with the methods

described in co-pending applications "COMPUTER BASED AUTOMATIC AUDIO MIXER" having Serial NoXX,XXXX, filed on December 27, 2000, assigned to Timbral Research Inc, hereby incorporated by reference in its entirety. and "ONLINE COMMUNICATION SYSTEM AND METHOD FOR AURAL STUDIES" having Serial No. XX/XXXX, filed on December 27, 2000, assigned to Timbral Research Inc, hereby incorporated by reference in its entirety. to implement a system which address the needs for automatic mixing and an online system for teachers and students to engage in aural studies, such as music and language.

SUMMARY OF THE INVENTION

The present invention provides a method and apparatus for the synchronization of an audio capture program and a streamed audio file, and for transmitting the captured audio data to a server. The synchronization program comprises the following components, (1) an audio stream rendering component capable of being controlled via a script, (2) a scripting component linked to a user interface, (3) an audio capture controller component that can receive the script

commands and issues directives to an audio capture engine, (4) an audio capture engine that communicates with an audio hardware subsystem of a personal computer, and (5) a data transmission component to communicate audio data to a server.

The audio stream rendering component must be capable of notifying other software components about event conditions or state changes immediately upon the occurrence of any one of the following events: (a) the beginning of a buffer stream, (b) the beginning of a rendered stream, and (c) completion of the rendering of a stream. The scripting component must be able to receive any of the above mentioned notification events from the audio stream rendering component.

The scripting component may further include the capability of scripting the audio capture controller component so that the scripting component can initialize, start, and stop the audio capture operation. The audio capture controller component must be able to respond to control commands issued from the scripting component. The audio capture controller component must also be capable of directing the audio capture engine, and provide a graphical user interface.

09750292.122700

The graphical user interface includes controls that allow a user to start and stop the audio rendering and audio capture engine. The graphical user interface also provides a status display to the user so that the user knows when the audio capture engine has been started and when it has been stopped, thereby alerting the user when his or her performance is being recorded.

Additionally, the audio capture engine may provide feedback information on the capture process. That is, if an error occurs during capture, a message should be displayed thereby notifying the users of the problem. Alternatively, if no errors are received the audio capture engine could display the amount of time remaining to stream the audio file, or the amount of available space on the client device to save the recorded file.

The synchronization program described above may be utilized in the following manner. A user initiates the synchronization program by launching the program on a client device. A graphical interface is presented to the user. The graphical interface allows the user to select from a plurality of audio files in which

09755062.122700

at least one track is missing. For example, a vocal or instrumental track may be missing thereby allowing the user to practice and record a vocal or instrumental performance. When the user selects an audio file, the user's input sends an event notification that is received by the scripting component and the file is streamed to the client from a server. The scripting component is initiated and instructs the audio rendering component to begin streaming the selected audio file, the scripting component also instructs the audio capture controller component to prepare for audio capture. When the audio stream rendering component transitions from a buffering state to a playback state, an event notification is sent from the audio stream-rendering component. The scripting component receives the event notification and immediately commands the audio capture controller to initiate audio capture. As the audio capture engine proceeds to capture the user's performance, the audio capture device notifies the audio capture controller component of the progress of the recording. The progress of the audio capture device may be displayed to the user in a graphical interface.

The user seeing this information may continue to record his or her performance or may choose to terminate the recording and reset the streamed audio file and the audio recording device.

Once the streamed audio file reaches its end, a third event notification is transmitted from the audio stream rendering component. The third event notification is received by the scripting component. Then the scripting component instructs the audio capture controller to stop audio capture.

Upon termination of the audio capture, the audio capture engine provides notification to the user through the graphical interface that the audio capture has stopped.

The beginning of the audio file recorded by the audio capture device is synchronized exactly with the beginning of the streamed audio file, thereby eliminating the need by the user to do any post processing to align the timing of the tracks.

In one embodiment of the present invention, the user is able to stream an audio file from a server device to a client device through the use of an audio stream player. This provides the user with the benefit of not having to download a large audio file from a server device to the client computer. Thereby the present invention allows the user to generate a recording almost instantly, without having to wait for the downloaded audio file. Furthermore, the streamed audio file is not stored on the client computer thereby freeing up resources on the client computer. Additionally, because the streamed file is not being saved on the client hard drive, there is less likelihood of the hard drive failures that result from transferring and manipulating large audio files.

The present invention also provides a mechanism for transmitting the performance captured on the client device to a server device for mixing. Performing the mixing process on the server rather than on the client is advantageous because a high quality version of the streamed audio accompaniment can be made available for mixing simply by storing it on the server's storage device. Also, the server itself can be a much more powerful computing engine than is available to typical users. The mix may be

accomplished on the server because the present invention times the start of the user's performance exactly with both the streamed audio file and its high quality mirrored version on the server, therefore the high quality version may be utilized without having to perform any time alignment of the two files.

Another benefit of transmitting the captured audio to the server for mixing is that expensive multi-track recording and mixing programs are not required. Furthermore, the user does not have to have the experience required to engineer a multi-track recording.

The synchronization method and its individual components will be described in greater detail below with reference to the included drawings. The invention further relates to machine readable media on which are stored embodiments of the present invention. It is contemplated that any media suitable for retrieving instructions is within the scope of the present invention. By way of example, such media may take the form of magnetic, optical, or semiconductor media. The invention also relates to data structures that contain embodiments of the present invention, and to the transmission of data structures containing embodiments of the present invention.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 Is a functional flow diagram illustrating an embodiment of the present invention.

FIG. 2 Is a functional flow diagram of the present invention illustrating additional steps.

FIG. 3 Is a function flow diagram illustrating an alternative embodiment of the present invention.

FIG. 4 is a function flow diagram of the alternative embodiment of the present invention illustrating additional steps.

DETAILED DESCRIPTION OF THE PRESENT INVENTION

Throughout this description reference will be made to a personal computer running Windows® and having associated components installed therein; one ordinarily skilled in the art shall understand that this is merely exemplary and other operating systems may be utilized with the present invention. Additionally, the present invention will be described in detail with regard to the user of a RealNetworks RealPlayer®; one of ordinary skill in the art shall understand that this description is merely exemplary and should not be considered limiting in any manner and that any other streaming audio program may be utilized. Furthermore, though the present invention will be described in detail as being performed between a client device and a server device operatively coupled for communication over a network, it shall be understood that the present invention may be performed on a client device independent of being operatively coupled to a server device.

Referring now to FIG. 1 there is shown a functional flow diagram of the present invention. At BOX 100, a user initiates the start of the present invention by logging into a server device from a client device operatively coupled for communication over a network. For example, the user may be utilizing a personal

computer connected to a server device utilizing the Internet. After logging into the server device the user is presented with a graphical interface or "web page" that provides a plurality of choices. For example, the user may be presented with a list of songs with which the user may perform. Alternatively, the user may be provided with only a single choice in the case of utilizing the present invention in a classroom environment. At BOX 100, the user selects an audio file to be streamed from the server device over a network connection.

At BOX 102, in response to the user selection received in BOX 100, the server transmits a capture control program (CCP) 90 and an audio capture program (ACP) 92 from the server device to the client device. If either the CCP 90 or the ACP 92 already exists on the client device from previous operation of the system, it is not downloaded again. CCP 90 is adapted to receive event conditions or state change notifications transmitted from an audio stream player and to provide start, pause and stop controls for the player within the web page. CCP 90 is further adapted to control the ACP 92 disposed within the client device.

09755062.122700

At BOX 105, the user initiates an audio stream player for playback of the selected audio file through the controls embedded in the web page.

At BOX 110, a first state change is transmitted from the audio stream player in response to the initiation of the audio stream player. For example, RealNetworks RealPlayer® transmits state changes, or event notifications in response to changes in the program's state. For example, when RealPlayer® changes from buffering to playback a state change is transmitted due to this transition.

At BOX 125, the audio file selected by the user begins to stream from a server device to the client device over a network and the audio stream player buffers data from the stream in memory on the client system until enough data are captured to ensure uninterrupted playback. Simultaneously, at BOX 120, CCP 90 prepares the ACP 92 for recording in response to the first state change transmitted in BOX 110. As shown in FIG. 1, the processes in both BOX 125 and BOX 120 are performed at the same time in response to the first state change that is transmitted by the audio stream player. Both processes must be completed prior to

09750602.122700

advancing to BOX 130. Typically the process of BOX 120 is completed much more rapidly than the process of BOX 125, so it is acceptable for the program to advance to BOX 130 upon completion of BOX 125.

At BOX 130, a second event condition is transmitted from the audio stream player in response to the state change of the audio stream player when it transitions from buffering to streamed playback.

At BOX 140, CCP 90 initiates the ACP 92, disposed on the client device, in response to the second state change transmitted from the audio stream player, and the audio capture is started.

At BOX 150, CCP 90 checks the status of the audio stream player to ensure that the audio file is streaming.

At Diamond 160, it is determined whether or not the user has stopped the audio stream player. If the player has not been stopped then the process of Diamond 165 is performed. If it is determined that the user has stopped the audio

stream player, then BOX 170 is performed. A user may wish to stop the audio stream player if they are not pleased with their performance, or if they are interrupted in the middle of their performance or for any number of other reasons.

At Diamond 165 it is determined whether the audio stream has been completed, that is if the end of the audio stream has been reached. If it is determined that the audio stream has not been completed then the process loops back to BOX 150 and is repeated as described above. If it is determined that the audio stream has been completed then BOX 170 is performed.

At BOX 170, a third state change is transmitted from the audio stream player in response to the player's transition from streaming to stopping.

At BOX 180, CCP 90 stops the audio capture ACP 92 in response to the third state change transmitted from the audio stream player.

Referring to FIG. 2 there is shown a functional block diagram of the method described above further including the steps shown in BOXES 185 and 190.

At BOX 185, the data captured on the client device by the audio capture program, ACP 92, is compressed using one of many commonly available programs. For example, the audio data captured and saved on the client computer may have been saved in an uncompressed format such as WAV. At BOX 185, the WAV file is compressed into a format such as MP3 or RM. The data is compressed so that the file uploads to the server device more rapidly. Also, if the captured audio file were stored on the client computer, it would utilize less storage space on the client device.

At BOX 190, the compressed audio file is uploaded to the server device over a network connection. For example, the client device may be coupled to the server device over the Internet.

Referring now to FIG. 3 there is shown a functional block diagram illustrating an alternative embodiment of the present invention.

Referring now to BOX 200, a user logs onto a server device from a client device that is operatively coupled for communication to the server device over a network. After logging onto the server device, the user is presented with a graphical display indicating at least one audio file that they may choose to download. The audio file may be missing at least one track that the user will perform, or alternatively the audio file may contain instructions for the user to follow. For example, if the audio file is a language lesson the file may contain prompts notifying the student when the student is to perform her or his part in the dialog. The audio file that was chosen by the student is downloaded from the server device to a storage device on the client device. The storage device may comprise any computer readable medium such as a hard drive, compact disc recordable/rewritable, or random access memory space.

At BOX 202, capture control program (CCP) 390 and audio capture program (ACP) 392 are transmitted from the server device to the client device.

CCP 390 is adapted to receive event conditions or state change notifications transmitted from an audio playback program. CCP 390 is further adapted to control audio capture ACP 392. If either CCP 390 or ACP 392 already exists on the client device from previous operation of the system, it is not downloaded again.

At BOX 203, the audio capture program, ACP 392, is prepared for recording.

At BOX 205, the user, having initiated an audio playback program on the client device, issues a command to the audio playback program to begin playing of the downloaded audio file.

At BOX 210, in response to requesting playback of the downloaded audio program, CCP 390 detects a first state change transmitted from the audio playback program.

At BOX 230, as the audio playback program begins playback of the downloaded audio file; a second state change is transmitted from the audio player.

At BOX 240, in response to the second state change transmitted from the audio playback device, CCP 390 directs ACP 392 to begin recording.

At BOX 250, CCP 390 checks the status of the audio playback program to ensure that the audio file is being played.

At Diamond 260 it is determined whether or not the user has stopped the audio player. If the player has not been stopped then the process of Diamond 265 is performed. If it is determined that the user has stopped the audio player, then BOX 270 is performed. A user may wish to stop the audio player if they are not pleased with their performance, or if they are interrupted in the middle of their performance or for any number of other reasons.

At Diamond 265, it is determined whether the audio playback has been completed, that is, if the end of the audio file has been reached. If it is determined

that the audio playback is not complete, then the process loops back to BOX 250 and is repeated as described above. If it is determined that the playback is complete, then BOX 270 is performed.

At BOX 270, a third state change is transmitted from the audio player in response to the player's transition from streaming to stopping.

At BOX 280, CCP 390 stops the ACP 392 in response to the third state change transmitted from the audio player.

Referring to FIG. 4 there is shown a functional block diagram of the method described above further including the steps shown in BOXES 285 and 290.

At BOX 285, the data captured on the client device by the audio capture ACP 392 is compressed using one of many commonly available programs. For example, the audio data captured and saved on the client computer may have been saved in an uncompressed format such as WAV. At BOX 285, the WAV file is

00275052.062302

compressed into a format such as MP3 or RM. The data is compressed so that the file uploads to the server device more rapidly. Also, if the captured audio file were stored on the client computer, it would utilize less storage space on the client device.

At BOX 290, the compressed audio file is uploaded to the server device over a network connection. For example, the client device may be coupled to the server device over the Internet.

Although the present invention has been described with reference to an implementation utilizing the main processor of a personal computer, it will be clear to those skilled in the art that it could be implemented as a dedicated hardware subsystem embedded in a personal computer. The hardware subsystem would take the form of a dedicated digital signal processing unit with the ACP and CCP instantiated in firmware, installed in either the computer's soundcard or built into a dedicated circuit board residing on the computer's data bus.

09750592.122700